

APR 11 2007

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

Remarks

In the foregoing amendment, claims 1-9, 36-38 and 42 have been canceled. Claim 17 was canceled previously. Pending in the application are claims 10-16, 18-35 and 39-41, of which claims 10, 16, 24, 26, 30, 34 and 35 are independent. The following comments address all stated grounds for rejection, and the Applicant respectfully submits that the presently pending claims, as identified above, are now in a condition for allowance. The rejections are traversed and reconsideration is requested.

I. Reopening of Prosecution

Prosecution was reopened in this case after Applicant filed a notice of appeal and a request for pre-appeal brief review. This appeal was from a final office action dated January 27, 2006, where the Examiner rejected claims 1-42 under 35 U.S.C. §102(e) as being anticipated by Suguta (U.S. Patent No. 6,901,579).

In this most recent office action of October 10, 2006, the Examiner explained the reasoning for reopening the prosecution as follows:

"A Pre-Appeal conference was held on September 26, 2006. It was decided that the claims can be better rejected under 37 [sic] U.S.C. 103(a) in light of the Suguta reference U.S. Patent no. 6,901,579. Prosecution is hereby re-opened."

II. Summary of Claim Rejections

Claims 1-9, 36-37 and 42 are rejected under 35 U.S.C. §112, second paragraph, as being indefinite.

Claims 16 and 18-23 are rejected under 35 U.S.C. §112, first paragraph as being unduly broad and/or 35 U.S.C. §112, second paragraph, as being indefinite.

Claims 1-16 and 18-42 are rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 6,901,579 ("Suguta").

These rejections will be discussed separately below.

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

III. Telephone Interview

Applicant thanks the Examiner for the courtesy of the telephone interview conducted on December 21, 2006.

During the interview, the Examiner alleged that the terms "critical" and "non-critical" in claims 1-9, 36-38 and 42 were indefinite because the claims did not define the terms. The Office Action ("OA") made no such indefiniteness rejection on this basis. Although Applicant disagrees with the Examiner, Applicant has canceled claims 1-9, 36-38 and 42 to expedite the prosecution of the present application.

The Examiner did not object to the terms "post-processing" and "core processing" in the remaining claims. Indeed, the phrases "post-processing" and "core processing" are defined at page 10, lines 4-16 of the present application and used throughout the specification of the present application. Hence, even if the Examiner had not stated his lack of objection, Applicant submits that the remaining claims are not subject to the alleged indefiniteness argument asserted against claims 1-9, 36-38, and 42.

IV. Claim Amendments

Applicant notes that canceling claims 1-9, 36-38, and 42 should not be construed as an acquiescence to the rejections in the Office Action. Rather, Applicant has canceled the claims to expedite the prosecution of the pending application. Applicant reserves the right to pursue the canceled claims in the present or continuation applications. Applicant submits that the claim amendments should be entered and considered.

V. Claim Rejections under 35 U.S.C. §112

A. Rejection of Claims 1-9, 36-37 and 42 under 35 U.S.C. §112, Second Paragraph

Claims 1-9, 36-37 and 42 are rejected under 35 U.S.C. §112, second paragraph as being indefinite. As claims 1-9, 36-37 and 42 have been canceled, Applicant respectfully submits that the rejection of claims 1-9, 36-37 and 42 is moot.

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

B. Rejection of Claims 16 and 18-23 under 35 U.S.C. §112, First Paragraph and/or Second Paragraph

It is not entirely clear from the Office Action whether these claims are being rejected under 35 U.S.C. §112, first paragraph, second paragraph or both. The Examiner states at page 3 of the office action,

“Claim 16 and 18-23 are rejected under 35 U.S.C. §112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.”

This language follows the quoting of the text of 35 U.S.C. §112, first paragraph. Moreover, this language is followed by:

“A single means claim, i.e., where a means recitation does not appear in combination with another recited element of means, is subject to an undue breadth rejection under 35 U.S.C. 112, first paragraph.”

Then, the Examiner asserts (at OA, page 4, lines 9-10) that:

“As per claim 16, claim 16 recites a single means, such as a graphical user interface to receive user inputs. This single means renders the claim indefinite.”

The Examiner notes from MPEP §2164.08, a single means claim is one where “a means recitation does not appear in combination with another recited element of means.” Claim 16, however, recites multiple elements and not a single means element. In addition, claim 16 has no means recitation. Claim 16 recites a system that includes a graphical user interface and an automatic code generator. As such, Applicant respectfully requests withdrawal of the rejection of claims 16 and 18-23 under 35 U.S.C. §112, first paragraph and/or second paragraph.

VI. Claim Rejections under 35 U.S.C. §103

Claims 1-16 and 18-42 are rejected under 35 U.S.C. §103(a) as being anticipated by U.S. Patent No. 6,901,579 (“Suguta”). Applicant respectfully traverses the rejection. The rejections

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

will be discussed below in the order of independent claims 1, 8, 9, 10, 16, 24, 26, 30, 34 and 35 and their dependent claims.

A. Claims 1-9, 36-38 and 42

Pending in the application are claims 10-16, 18-35 and 39-41. Applicant respectfully submits that the rejection of claims 1-9, 36-38 and 42 is moot.

B. Claims 10 and 26

Claim 10 reads as follows:

A computer implemented method comprising:
specifying a model, the model including sections, a first subset of the sections designated post-processing unit sections and a second subset of the sections designated as core processing unit sections; and
generating software source code for the model with a code generator using the second subset.

Claim 26 reads as follows:

A computer program product residing on a computer readable medium having instructions stored thereon which, when executed by the processor, cause the processor to:
specify a model, the model including sections, a first subset of the sections designated post-processing unit sections and a second subset of the sections designated as core processing unit sections; and
generate software source code for the model with a code generator using the second subset.

1. "a model, the model including sections, a first subset of sections designated post-processing unit sections and a second subset of sections designated as core processing unit sections"

Sugata fails to disclose or suggest "a model, the model including sections, a first subset of the sections designated post-processing unit sections and a second subset of the sections designated as core processing unit sections," as required by claims 10 and 26. Sugata is entirely silent as to a model having post-processing unit sections and core processing unit sections. Moreover, as will be discussed in more detail below, Sugata does not teach or suggest models, and indeed does not mention such a term, because Sugata is not concerned with models.

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

Suguta discloses an approach for automatically generating a program or a portion of a program in an object-oriented programming language (column 1, lines 11-14). To that end, Suguta discloses storing object class definitions 11 (see Figure 3) in a class definition storage unit 1 (column 5, lines 13-15). Figure 5 shows an example of a class definition. The internal structure of each class definition 11 is parsed by a parsing unit 2 to generate parsing information 31 (column 5, lines 15-17). Figure 6 shows an example of the parsing information that results from parsing the class definition of Figure 5. A program or a portion of a program is generated using the parsing information and generation patterns descriptions 41, which are stored in the patterns description storage unit 4 (column 5, lines 20-21). Each generation pattern description 41 describes a relationship between a plurality of object classes and is used for generating program patterns (col. 2, lines 52-57 and 62-64). In short, Suguta addresses code generation from class definitions using predefined generation patterns.

In formulating the rejection of claim 10, the Examiner relies upon the arguments set forth in the Office Action (page 7) regarding claims 1, 8 and 9. In particular, the Examiner states: "As per claims 10-15, most of the limitations of these claims have been note [sic] in the rejection of claims 1, 8 and 9 above . . ." Similarly, as to claim 26, the Examiner refers to his early statements, "As per claims 24-29, all the limitations of these claims have been noted in the rejection of claims 1-23 above." (OA page 8).

Applicants, thus, have reviewed the arguments made with rejection to claims 1, 8 and 9, even though these claims have been cancelled. At pages 6 and 7 of the Office Action, the Examiner notes with respect to claims 1, 8 and 9 (emphasis in the original):

Although Suguta discloses generation of source codes including *means for indicating the role of the class in the generated source code (See Suguta Title)*; it is noted, however, Suguta did not specifically detail the aspects of *model that are critical or non critical to a real time execution* as recited in the instant claims. However, One of ordinary skill in the art would have found it obvious, at the time of the invention, in the source code generation system of Suguta, that aspects that are critical or non critical is realized because when the system of Suguta is generating a source code, functions (although being part of the

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

overall software) that are important to realized [sic] the source code generation are activated and functions (although being part of the overall software) that are not necessary for the generation of the source is [sic] not called or activated. In general, when a computer application or a computer operating system is being initiated, the modules that are necessary for the initiation of the software program is called; whereas, modules that are not needed for the computer program to initiate are not called. Further, the means for indicating the role of the class in the generated source code (*See Suguta Title*) disclose [sic] by Suguta is an indication that Suguta realized that it is important for its system to understand the role of each module in the overall source code generated so that each module can be called for its proper or appropriate role during execution of the source code. Therefore, the claimed feature of generating source code and identifying portions that are critical or non-critical can be realized during execution of the source code of Suguta since the determination of what is critical or non-critical to achieve execution of the generated source code is necessary and will be achieved in the background during execution of the generated source code.

From this language, it is evident that the Examiner acknowledges that Suguta does not disclose a model having a first set of sections designated as post-processing sections and a second subset of sections designated as core-processing sections.

In the above-quoted language, the Examiner argues that it would have been obvious to one skilled in the art at the time of the invention to detect what aspects of the model are critical or non-critical to real time execution. As support for this position, the Examiner notes that one skilled in the art would have found it obvious that in the Suguta system, functions that are imported to be realized source code generation are activated and functions that are not necessary for source code generation are not activated. The Examiner also notes more generally that when a computer application or operating system is being initiated, the values necessary for the initiation are called and values that are not necessary for the initiation are not called. The Examiner concludes in view of these statements that the claimed features will be realized in the background during execution of the generated source code.

Assuming *arguendo*, that the statements put forth by the Examiner are true, claim 10 still would not be obvious to one of ordinary skill in the art at the time of the invention. Speaking

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

broadly, there are at least three distinct entities that may be involved in code generation: (1) software that is used to generate code, (2) a description for which the code is to be generated, such as a model of a system in the present application, and (3) code that is produced through code generation.

As far as Applicant can determine, all of the Examiner's remarks regarding Suguta's code generation cited above or general remarks regarding workings of "a computer application or a computer system [that] is being initiated" address the internal workings of entity (1) – the software that is used to generate code. However, none of the presently pending claims address the internals of code generation software, except as they can be interpreted to do so in specifying how entity (3) – the code – is to be generated from entity (2) – the model. The "sections" recited in claims 10 and 26 are not "functions" that are activated or not activated to realize code generation, as stated by the Examiner. These sections are sections of the model, not of the code generation software. Consequently, these sections are considered by the code generated software to generate resulting source code.

That is, claim 10 is not concerned with what functions are to be called during code generation like the aspects of the Suguta system to which the Examiner points, but rather is concerned with what subset of sections of the model are designated as core-processing unit sections so that code will be generated for those designated core-processing unit sections of the model. Neither Suguta, nor the Examiner's remarks on the general nature of initiation of software applications or operating systems render claims 10 or 26 as obvious.

Moreover, claims 10 and 26 require the affirmative step of specifying a model having a first subset of sections designated as post-processing unit sections and a second subset of sections designated as core-processing unit sections. The claim requires that the model be specified and include designated sections. In the aspects of the Suguta system identified by the Examiner, there is no designation in the model of a subset of sections of the model as core processing unit sections and there is no designation in the model of a subset of sections as post-processing unit sections.

APR 11 2007

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

In addition, there is no recognition in Suguta and no demonstration that there was a recognition of one skilled in the art at the time of the invention that a model be specified to have a first subset of sections designated as post-processing unit sections and a second subset of the sections designated as core processing unit sections. Suguta is entirely silent with respect to these features of the claimed invention. It appears that the Examiner is employing the teachings of the present application to create an obviousness argument. This is not permissible.

Applicant wishes to emphasize that Suguta does not mention "models" anywhere. It is concerned with generating object oriented program instructions from class definitions and not models.

For at least the foregoing reasons, Applicant respectfully urges that the rejections of claims 10 and 26 be withdrawn and the claims be allowed.

2. "generating" or "generate" "software source code for the model with a code generator using the second subset"

Applicant respectfully submits that Suguta does not teach "generating" or "generate" "software source code for the model with a code generator using the second subset," as required by claims 10 and 26. In claims 10 and 26, code for the model is generated using the core processing unit sections.

In contrast, Suguta teaches the automatic generation of at least a portion of an object-oriented program. (See Suguta, Abstract). In the generation, Suguta uses a class definition and a generation pattern description provided by users. (See Suguta, Column 6, Lines 30-38). In Suguta, the class definition and the generation pattern description do not include a first subset of sections designated post-processing unit sections and a second subset of sections designated as core processing unit sections, as required by claims 10 and 26. Suguta also does not teach generating software source code for a model using the second subset of sections designated as core processing unit sections, as required by claims 10 and 26.

For at least the foregoing reasons, Applicant respectfully requests that the rejection of claims 10 and 26 be withdrawn and the claims be allowed.

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

C. Claims 11-15 and 39

Claims 11-15 and 39 depend on claim 10 and, as such, incorporate all of the features of claim 10. Accordingly, claims 11-15 and 39 are not rendered obvious over the cited reference for at least the reasons stated above in connection with claim 10. Applicant respectfully requests that the rejection of claims 11-15 and 39 be withdrawn and the claims be allowed.

Furthermore, Applicant respectfully submits that Suguta fails to teach or suggest all of the limitation recited in claim 11. Specifically, Applicant submits that Suguta does not teach "the post-processing unit sections are logical units of the model that have no data outputs that feed core processing unit sections."

Suguta also fails to teach or suggest all the limitations of claims 12 and 15. Applicant submits that Suguta does not respectively teach "linking the code to the first subset of sections through an inter-process communication link" and "executing the code on a target processor," as required dependent claim 12, and "receiving output from the code via the inter-process communications link" and "processing the output in the first subset," as required dependent claim 15. The Examiner points (OA page 8) to the following language in Suguta at column 6, lines 30-38 as teaching these features:

In FIG. 4, an input/output unit 9 is connected to a computer system 8. A user (a programmer) can give an instruction such as an a class definition 11 or a generation pattern description 41 to the computer system using a keyboard, etc., so that a program is generated, and the result of a process is obtained from the computer system, or the result of a generation of a message or a program based on an output from a display, etc. of the input/output unit, is obtained.

Applicant respectfully disagrees. Suguta teaches in Fig. 4, an input/output unit (9) connected to a computer system (8). Suguta also teaches that the computer system (8) includes a CPU (81) connected to a storage unit (82). These connections are not an inter-process communication link recited in claims 12 and 15. Since the computer system (8) disclosed in Suguta is not a multi-process system, Suguta does not need an inter-process communication link, as required by claims 12 and 15. Suguta does not teach a target processor for executing generated code, as required by claim 12. Moreover, were Suguta to be interpreted to teach the

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

inter-process communication link, *arguendo*, the rejection would still be improper, because Suguta fails to teach "linking linking the code to the first subset of sections through an inter-process communication link." As discussed above, nothing in Suguta teaches sections of the model, but even if it did, there is absolutely no discussion of linking a subset of the model sections to the generated code using the inter-process communication link.

As such, Applicant respectfully requests that the rejection of claims 11-15 and 39 be withdrawn and the claims be allowed.

D. Claims 27-29

Claims 27-29 depend on base claim 26 and, as such, incorporate all of the features of claim 26. Accordingly claims 27-29 are not rendered obvious for the reasons set forth above with respect to claim 26. Applicant respectfully requests that the rejection of claims 27-29 be withdrawn and the claims be allowed.

E. Claim 16

Applicant respectfully submits that Suguta fails to teach or suggest all of the features of claim 16. Specifically, Applicant submits that Suguta does not teach at least the following two elements: "a GUI that is adapted to receive user inputs to specify components of a model in one of a first subset of sections designated as post-processing elements of a model and a second subset of sections designated as core elements of the model" and "an automatic code generator for generating code capable of real-time execution based on the second subset of sections," as required by claim 16. These claim elements will be discussed in sequence below.

1. "a GUI that is adapted to receive user inputs to specify components of a model in one of a first subset of sections designated as post-processing elements of a model and a second subset of sections designated as core elements of the model"

Applicant submits that Suguta does not teach "a GUI that is adapted to receive user inputs to specify components of a model in one of a first subset of sections designated as post-processing elements of a model and a second subset of sections designated as core elements of the model," as required by claim 16. The Examiner points (OA page 8) to the following language in Suguta at column 6, lines 30-56 as teaching these features:

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

In FIG. 4, an input/output unit 9 is connected to a computer system 8. A user (a programmer) can give an instruction such as an a class definition 11 or a generation pattern description 41 to the computer system using a keyboard, etc., so that a program is generated, and the result of a process is obtained from the computer system, or the result of a generation of a message or a program based on an output from a display, etc. of the input/output unit, is obtained.

The computer system includes a CPU 81 for performing operations, and a storage unit 82. The CPU 81 reads out a program 83 required for performing a process from the storage unit 82, and executes the program 83 according to the instruction given by the input/output unit 9. Furthermore, the CPU provides the data 1, 3, and 4 stored in the storage unit 82 to the program 83, and generates the generated program 7. The contents of the class definition storage unit 1 are applied to parsing unit 2 and the processed result stored in the parsing information storage unit 3, the contents of the generation pattern description storage unit 4 and the parsing information storage unit 3 are applied to program generation unit 5, and the processed result of the program generation unit 5 is further processed in the generation verification unit 6, and if there is no conflict definition in the processed result, the result is stored in the generated program 7. If the generation verification unit 6 finds some conflict, an alarm signal is generated for the user.

Suguta teaches in Fig. 4 an automatic object-oriented program generation apparatus including an input/output unit (9). Suguta also teaches that a user (a programmer) can give an instruction such as a class definition (11) or a generation pattern description (41) to the computer system so that a program is generated. The user interface in Suguta appears to be adapted to receive a class definition (11) or a generation pattern description (41) from a user. Suguta, however, does not teach a graphical user interface that enables a user to specify which components of a model are post-processing elements or core elements of the model, as required by claim 16.

2. "an automatic code generator for generating code capable of real-time execution based on the second subset of sections"

Applicant submits that Suguta does not teach "an automatic code generator for generating code capable of real-time execution based on the second subset of sections," as required by claim 16. Suguta teaches a program generation unit (5) in Figure 3. In Suguta, the program generation unit (5) generates an object oriented language program using information on the internal structure of an input class definition (11) and a generation pattern description (41).

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

Suguta, however, does not teach or suggest an automatic code generator for generating code capable of real-time execution.

Applicant respectfully submits that Suguta does not teach generating code that is capable of real-time execution based on the second subset of sections. The Examiner points (OA, page 6) to the following language in Suguta at column 2, lines 40-51 as teaching this feature.

A first object of the present invention is to enable automatic generation of a copy constructor in an object-oriented programming language program which enables duplication of an object. Conventionally, such a program as shown in FIG. 2C must be made by a user. A second object of the present invention is to enable verification of whether or not an automatically generated program would include a definition which conflicts with that of an existing program when the program is generated. A third object of the present invention is improving the productivity of program development by enabling automatic generation of an object-oriented program.

The Examiner asserts in the Office Action that "since Suguta allows automatic generation of source codes, generation of code that is capable of real-time execution is realized." (See the Office Action, page 6, lines 9-11). Applicant respectfully disagrees.

Although Suguta teaches automatic generation of an objected oriented language program, Suguta does not teach generation of code for a model that is capable of real-time execution based on the second subset of sections designated as core elements of the model, as required by claim 16. There are no disclosures, teachings and suggestions of real-time execution of generated source code in Suguta. Suguta is unconcerned with code execution and does not teach real-time execution of generated source code. Suguta does not teach or suggest generating a program capable of real-time execution.

In light of the aforementioned arguments, Applicant respectfully requests that the rejection of claim 16 under 35 U.S.C. §103(a) be withdrawn and the claim be allowed.

F. Claims 18-23 and 40-41

Claims 18-23 and 40-41 depend on base claim 16 and, as such, incorporate all of the features of claim 16. Accordingly claims 18-23 and 40-41 are not rendered obvious for the

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

reasons set forth above with respect to claim 16. Applicant respectfully requests that the rejection of claims 18-23 and 40-41 be withdrawn and the claims be allowed.

Furthermore, Applicant respectfully submits that Suguta does not teach "a link to provide an inter-process communication between the code and the first subset of sections," as required dependent claim 19.

In rejecting claim 19 (OA page 8), the Examiner relies upon the arguments set forth in the OA regarding claims 1, 8, 9 and 10-15. In particular, the Examiner stated, "As per claims 16, 18-23, most of the limitations of these claims have been note [sic] in the rejection of claims 1, 8, 9 and 10-15 above . . ." The Examiner also states that Suguta discloses essential computer components in Figure 4. Suguta teaches an input/output unit (9) connected to a computer system (8). Suguta also teaches that the computer system (8) includes a CPU (81) connected to a storage unit (82). These connections are not an inter-process communication link recited in claim 19. Since the computer system (8) disclosed in Suguta is not a multi-process system, Suguta does not need an inter-process communication link, as required by claim 19.

In light of the aforementioned arguments, Applicant respectfully requests that the rejection of claims 11-15 and 39 be withdrawn and the claims be allowed.

G. Claims 24, 34 and 35

Applicant respectfully submits that Suguta fails to teach or suggest all of the features of claims 24, 34 and 35. Specifically, Applicant submits that Suguta does not teach at least the following features: "receiving" or "receive" "user input through a graphical user interface (GUI) specifying a block diagram model, the block diagram model including sections, a first subset of sections designated post-processing unit sections and a second subset of the sections designated as core processing unit sections"; "generating" or "generate" "software source code for the block diagram model with a code generator using the second subset"; and "connecting" or "connect" "the software source code to the first subset via an inter-process communication link." These claim features will be discussed in sequence below.

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

1. "receiving" or "receive" "user input through a graphical user interface (GUI) specifying a block diagram model, the block diagram model including sections, a first subset of sections designated post-processing unit sections and a second subset of the sections designated as core processing unit sections"

Applicant respectfully submits that Suguta does not teach "receiving" or "receive" "user input through a graphical user interface (GUI) specifying a block diagram model, the block diagram model including sections, a first subset of sections designated post-processing unit sections and a second subset of the sections designated as core processing unit sections," as required by claims 24, 34 and 35.

Suguta teaches the generation of an object oriented program from a class definition and a generation pattern description. (See Suguta, abstract). Suguta teaches a user interface for the inputs of a class definition and a generation pattern description. (See Suguta, column 6, lines 30-38). The Examiner relies upon the arguments set forth in the OA regarding claims 1-23 in order to reject claim 24 (OA page 8). The Examiner also relies upon the arguments set forth in the OA regarding claims 1-33 in order to reject claims 34 and 35 (OA page 9). In addition to the arguments set forth above against the rejection of claims 10, 16 and 26, Applicant submit that Suguta at least does not teach *receiving* user input through a graphical user interface (GUI) *specifying a block diagram model*.

2. "generating" or "generate" "software source code for the block diagram model with a code generator using the second subset"

Applicant respectfully submits that Suguta does not teach "generating" or "generate" "software source code for the block diagram model with a code generator using the second subset," as required by claims 24, 34 and 35.

Suguta teaches the generation of an object oriented program from a class definition and a generation pattern description. (See Suguta, abstract). Suguta teaches a user interface for the inputs of a class definition and a generation pattern description. (See Suguta, column 6, lines 30-38). The Examiner relies upon the arguments set forth in the OA regarding claims 1-23 in order to reject claim 24 (OA page 8). The Examiner also relies upon the arguments set forth in the OA regarding claims 1-33 in order to reject claims 34 and 35 (OA page 9). In addition to the

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

arguments set forth above against the rejection of claims 10, 16 and 26, Applicant submit that Suguta does not teach *generating* software source code *for a block diagram model*. Suguta is irrelevant to a block diagram model.

3. "connecting" or "connect" "the software source code to the first subset via an inter-process communication link"

Applicant respectfully submits that Suguta does not teach "connecting" or "connect" "the software source code to the first subset via an inter-process communication link," as required by claims 24, 34 and 35. As set forth above against the rejection of claims 12, 15 and 19, Applicant submits that Suguta does not teach an inter-process communication link that connects the software code, which is generated using a second subset of sections designated as core processing unit sections, to the first subset of sections designated post-processing unit sections. Since the system disclosed in Suguta is not a multi-process system, Suguta does not need to teach an inter-process communication link.

In light of the aforementioned arguments, Applicant respectfully requests that the rejections of claims 24, 34 and 35 under 35 U.S.C. §103(a) be withdrawn and the claims be allowed.

H. Claim 25

Claim 25 depends on base claim 24 and, as such, incorporates all of the features of claim 24. Accordingly claim 25 is not rendered obvious for the reasons set forth above with respect to claim 24. Applicant respectfully requests that the rejection of claim 25 be withdrawn and the claim be allowed.

I. Claim 30

Applicant respectfully submits that Suguta fails to teach or suggest all of the features of claim 30. Specifically, Applicant submits that Suguta does not teach at least the following two features: "specify a block diagram model, the block diagram model including data having internal pre-defined data storage classes and external custom data storage classes"; and "generate software source code for the block diagram model with a code generator using the internal predefined data storage classes and the external custom data storage classes." The Examiner

Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

alleges (OA, page 8) that Suguta teaches specifying a block diagram model at the title and abstract. The Examiner also alleges (OA, page 9) that Suguta teaches generating software code for the block diagram at column 2, line 40 – column 3, line 19. Suguta teaches the generation of an object oriented program from a class definition and a generation pattern description. (See Suguta, abstract and column 2, line 40 – column 3, line 19). Suguta, however, does *not* teach specifying a block diagram model, and generating software source code for the block diagram model. Suguta does not disclose a block diagram model at all.

In light of the aforementioned arguments, Applicant respectfully requests withdrawal of claim 30 under 35 U.S.C. §103(a).

J. Claims 31-33

Claims 31-33 depend on base claim 30 and, as such, incorporates all of the features of claim 30. Accordingly claims 31-33 are not rendered obvious for the reasons set forth above with respect to claim 30. Applicant respectfully requests withdrawal of the rejection of claims 31-33.

VII. Conclusion

In light of the aforementioned arguments, Applicant submits that Suguta fails to disclose, teach or suggest the patentable features of the invention, and contends that the claimed invention is novel and non-obvious in view of Suguta.

Please charge any shortage or credit any overpayment of fees to our Deposit Account No. 12-0080, under Order No. MWS-041. In the event that a petition for an extension of time is required to be submitted herewith, and the requisite petition does not accompany this response, the undersigned hereby petitions under 37 C.F.R. §1.136(a) for an extension of time for as many months as are required to render this submission timely. Any fee due is authorized to be charged to the aforementioned Deposit Account.

In view of the above comments, Applicant believes that the pending application is in condition for allowance and urges the Examiner to pass the claims to allowance. Should the

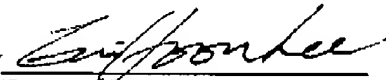
Application No.: 09/910,170
Art Unit: 2161

Docket No.: MWS-041

Examiner feel that a teleconference would expedite the prosecution of this application, the
Examiner is urged to contact the Applicant's attorney at (617) 227-7400.

Dated: April 11, 2007

Respectfully submitted,

By 

EuiHoon Lee

Registration No. L0248

LAHIVE & COCKFIELD, LLP

One Post Office Square

Boston, Massachusetts 02109

(617) 227-7400

(617) 742-4214 (Fax)

Attorney for Applicant